FSF3847
Convex Optimization
with Engineering Applications
Assignment 4

Franco Ruggeri

May 9, 2023

1 Exercise 4.1

Part (a)

Let $f \coloneqq -\sum_{i=1}^{m} \log(1 + x_i c_i)$, $g_i(x) \coloneqq b_i - a_i^T x, i = 1, ..., n$, $g_i(x) \coloneqq -x, i = n+1, ..., n+m$, $l \coloneqq n+m$. The max sum log problem can be written as:

$$\begin{aligned} &\text{minimize} \quad && f(x) \\ &\text{subject to} \quad && g(x) \succeq 0 \end{aligned}$$

The Lagrangian is:

$$L(x, y) = f(x) - y^T g(x)$$

Since $c \in \mathbb{R}_+^m$, the domain of the objective function $\mathbf{dom} f$ agrees with the positivity constraints $x \succ 0$:

$$\mathbf{dom} f = \{x \mid 1 + x_i c_i > 0\} = \{x \mid x_i > -\frac{1}{c_i}\} \supset \{x \mid x_i > 0\}$$

Furthermore, since $A \in \mathbb{R}_+^{n \times m}$, and $b \in \mathbb{R}_{++}^n$, we have:

$$\lim_{x_j \to 0^+} \underbrace{b_i}_{>0} - \underbrace{a_{ij}}_{\geq 0} x_j = b_i > 0$$

Thus, it is possible to select a small-enough $x \succeq 0$ ($x \approx 0$) that is a feasible interior starting point.

I chose to implement a primal-dual interior method, whose pseudo-code is described in Alg. 1. The line search is executed with a simple backtrace search that reduces the step length until the new point is primal-dual strictly feasible,

Algorithm 1: Primal-dual interior method.

Input: starting interior feasible point $\begin{bmatrix} x & y \end{bmatrix}^T$, hyperparameters σ, β, ϵ

repeat

  // Reduce barrier parameter

  $\mu \leftarrow \sigma \frac{\hat{\eta}}{l}$

  // Compute search direction (Newton equations)

  $A_{newton} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = b_{newton}$

  // Line search (with backtrace)

  $\alpha \leftarrow 1$

  while *Infesible*$(x + \alpha \Delta x,\ y + \alpha \Delta y)$ **do**

    $\alpha \leftarrow \beta \alpha$

  end

  $x \leftarrow x + \alpha \Delta x$

  $y \leftarrow y + \alpha \Delta y$

  // Compute surrogate duality gap

  $\hat{\eta} = g(x)^T y$

until $\hat{\eta} \leq \epsilon$

starting from the unit step length. In the following, the Newton equations for the specific max sum log problem are derived.

The general Newton equations for a convex program *without equality constraints* are (from Lecture 9 and [1]):

$$\begin{bmatrix} \nabla^2_{xx} L(x,y) & A_{\mathcal{I}}(x)^T \\ Y A_{\mathcal{I}}(x) & -G(x) \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta y \end{bmatrix} = - \begin{bmatrix} \nabla_x L(x,y) \\ G(x)y - \mu \mathbf{1} \end{bmatrix}$$

where $G(x) = \mathbf{diag}(g(x))$, $Y = \mathbf{diag}(y)$, and $A_{\mathcal{I}}(x) = \begin{bmatrix} \nabla g_1(x) & \ldots & \nabla g_l(x) \end{bmatrix}^T$.

Table 1: Results of the implemented primal-dual interior method (Alg. 1) and a CVX solver for some randomly generated problems.

Dimensions		Relative error			Time (ms)		Iter.	$x_j \neq 0$
m	n	x	y	$f(x)$	Alg. 1	CVX	Alg. 1	Alg. 1
10	2	$2.1 \cdot 10^{-6}$	$5.7 \cdot 10^{-6}$	$3.8 \cdot 10^{-7}$	5.2	5.6	13	4
10	15	$2.5 \cdot 10^{-6}$	$5.6 \cdot 10^{-6}$	$1.4 \cdot 10^{-6}$	5.9	5.7	13	3
100	5	$6.9 \cdot 10^{-6}$	$3.1 \cdot 10^{-6}$	$9.2 \cdot 10^{-8}$	16.9	8.9	14	68
100	20	$3.3 \cdot 10^{-5}$	$3.3 \cdot 10^{-5}$	$5.8 \cdot 10^{-7}$	40.7	26.1	19	16

For the specific max sum log problem, we have:

$$\nabla_x L(x,y) = \nabla f(x) - \sum_{i=1}^{l} y_i \nabla g_i(x) = \nabla f(x) - A_{\mathcal{I}}(x)^T y$$

$$\nabla^2_{xx} L(x,y) = \nabla^2 f(x) - \sum_{i=1}^{l} y_i \nabla^2 g_i(x)$$

$$\nabla f(x) = -\begin{bmatrix} \frac{c_1}{1+c_1x_1} & \cdots & \frac{c_m}{1+c_mx_m} \end{bmatrix}^T$$

$$\nabla^2 f(x) = \mathbf{diag}\left(\begin{bmatrix} (\frac{c_1}{1+c_1x_1})^2 & \cdots & (\frac{c_m}{1+c_mx_m})^2 \end{bmatrix}\right)$$

$$\nabla g_i(x) = -a_i,\ i = 1, ..., n$$

$$\frac{\partial g_i(x)}{\partial x_j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases},\ i = n+1, ..., l$$

$$\nabla^2 g_i(x) = 0$$

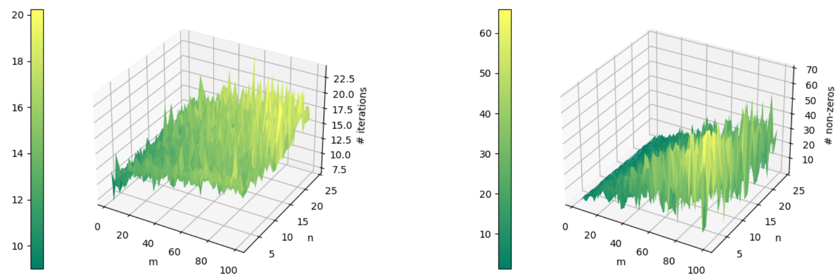
$$A_{\mathcal{I}} = \begin{bmatrix} -A \\ \mathbf{I}_{m\times m} \end{bmatrix}$$

Part (b)

The implementation has been tested for some randomly generated problems with different dimensions. In the numerical experiments, the hyperparameters were set to $\sigma = 0.1$ (long-shot method), $\beta = 0.8$ (from [1]), and $\epsilon = 10^{-5}$. The results, reported in Table 1, show the comparison between the solutions computed by the implemented algorithm and the solutions computed by the reference CVX solver. The relative errors are smaller than 10^{-5}.

Part (c)

Figure 1 shows the number of iterations and number of non-zeros elements in x for more problems. It is possible to identify some trends. First, the number



(a) Iterations (b) Non-zeros

Figure 1: Number of iterations (a) and non-zeros elements of x (b) for varying dimensions m (number of variables) and n (number of constraints).

of iterations tends to increase as the dimensions increase. This result makes sense, as increasing n and/or m complicates the problem. Second, the number of non-zeros in x seems to increase with m but slightly decrease with n. A possible interpretation is that, given a fixed m (number of variables), adding other constraints $a_i^T x < b_i$ (i.e., increasing n) can move the optimal solution from active constraints $x_j = 0$ to new active constraints $a_i^T x = b_i$. On the contrary, given a fixed m, removing some active constraints $a_i^T x < b_i$ (i.e., decreasing n) can move the optimal solution to other active constraints $x_j = 0$.

References

[1] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.